REMARKS

In a final office action dated May 2, 2007, the Examiner objected to the Drawing; objected to the Specification; and rejected claims 1-5, 12-14, 16, 18 and 20-28 under 35 U.S.C. 102(e) as anticipated by Chiles et al. (U.S. Patent 6,16,567).

In response to the Drawing objection, applicants herewith submit a replacement drawing sheet for Fig. 3. No new matter is introduced.

In response to the objection to the Specification, applicants have amended the Specification to spell out the first occurrence of the acronym "EDVAC". No new matter is introduced.

Applicants believe that the pre-existing claims sufficiently distinguish over the cited art, in particular because *Chiles* discloses a "script" which is essentially a list of files to be downloaded and/or executed, rather than instructions of an instruction set to be compiled and executed.

However, in the interests of furthering prosecution herein, applicants have amended all independent claims to further define the scope of the present invention. These amendments are made without conceding that the existing claims are not patentable over *Chiles*, and applicant reserves the right to pursue these and other claims in one or more continuations or divisional patent applications, or to re-assert these claims in the present application in the event of further prosecution herein.

In particular, the claims are amended to recite use of multiple upgrade (or installation) objects each having a respective sequence number, wherein at least one object has a prerequisite, the prerequisite allowing it to be executed out of sequence. Various dependent claims have been

amended to conform to the amendments to the independent claims. Claims 3, 20, 27 and 28 have been cancelled as superfluous.

Support for the additional subject matter is found is several places. Fig. 3 and the description thereof at p. 9, line 13 - p. 10, line 18, discloses sending a group of objects. Fig. 2 and accompanying text at p. 8, line 16 - p. 9, line 6 discloses the use of sequence numbers and prerequisites in the objects, and that the prerequisites may be used to allow some steps to be run out of order. No new matter is introduced.

As explained previously, applicants' invention is intended to simplify software maintenance in a complex enterprise environment, in which many copies of software are supported on different systems, each having different hardware and/or software configuration. It is difficult to guarantee that installation or upgrade prerequisites will be met on all such systems, that configurations will be the same, and so forth. Conventionally, most businesses employ persons who are trained to deal with all the possible contingencies and who install software on the applicable systems. This is a very labor intensive approach. It is also theoretically possible to create custom software for installation, but again the cost of developing such software is high.

Applicants' invention involves an improved upgrade process, whereby a centralized system administrator or similar person creates upgrade or installation objects in the form of simple scripts. These scripts specify some series of actions to be taken to install or upgrade the code. The scripts are contained within upgrade or installation objects, which specify any required prerequisites for script execution and other information. Multiple objects, each having its own sequence number and prerequisites, if any, support a single installation or upgrade, thus facilitating flexibility in design and ease of re-use of installation or upgrade objects. The "scripts" are not executable programs, but require the use of a separate script processor for execution. The

script processor is distributed in advance, preferably as part of the computer program, but only need be distributed once, and is a permanent part of each target system. The script processor is executable code which reads the script contained in the update object and performs the upgrade according to the script. I.e., the script processor is essentially a compiler or interpreter which "compiles and executes" the script commands to performs the installation or upgrade.

Chiles discloses an upgrade technique, in which a software upgrading application is loaded on a client system, and an "update script" is downloaded from a server. The "upgrade script" is essentially a list of files to be downloaded, with or without execution. The software upgrading application receives the "update script", downloads the listed files, and invokes execution of any files indicated to be "run". The "update script" may contain a version identifier, which is compared to the existing version to verify that the version being downloaded is more recent than the currently existing version on the system. .

Applicants' representative claim 1, as amended, recites:

1. A method of upgrading a computer program installed on a first computer system, the computer program including a script processing module, the method comprising:

receiving *a plurality of upgrade objects* associated with the computer program, each said upgrade object including a respective script comprising a plurality of *script instructions of a pre-defined script instruction set* adapted for use by the script processing module to upgrade the computer program, the script being not independently executable without the script processing module, each said upgrade object being generated on a second computer system remote from said first computer system and transmitted from said second computer system to said first computer system, wherein each said upgrade object contains a respective sequence number, *wherein at least one said upgrade object contains at least one respective prerequisite, and wherein not all said upgrade objects have the same at least one respective prerequisite*;

with respect to each said upgrade object containing at least one respective prerequisite, automatically determining whether the at least one respective prerequisite has been met; and

performing an upgrade of said computer program by *compiling and executing each said script* on said first computer system *with the script processing module*, wherein, with

respect to each said upgrade object containing at least one respective prerequisite, the respective script contained in the object is not compiled and executed until the respective at least one prerequisite contained in the object has been met, and *wherein the at least one prerequisite allows at least one script to be executed out of sequence*. [emphasis added]

Independent claims 12 and 16 contain analogous limitations to the italicized limitations above. Independent claims 18 and 23 recite installing the program rather than upgrading it, the script processor being part of the environment to which the program is installed, but otherwise contain analogous recitations to the italicized limitations above.

As explained above, *Chiles* discloses sending an "update script" which amounts to nothing more than a list of modules to be downloaded and/or executed. Although the list does indeed cause the software updating application to do something, what it does is download and (optionally) run the downloaded modules. There is no disclosure of "compiling and executing" instructions of a pre-defined instruction set contained in the script.

Nor is there any disclosure of multiple script objects. Although a number is sometimes considered arbitrary, the use of multiple separate script objects serves a purpose in applicant's technique. It enables prerequisite requirements to be evaluated separately for the different script objects, allowing greater flexibility of ordering operations. In *Chiles*' scheme, there is simply no incentive to use multiple update scripts. The update script being just a simple list, multiple upgrade files will not accomplish anything more than a single file.

Finally, there is no disclosure of "prerequisites", as used by applicants. *Chiles* does indeed disclose use of a version number, which the Examiner likens to a prerequisite. I.e., *Chiles*' update script lists a version number of the update; the software application checks the existing version, and does not download unless the existing version is older than the upgrade. This avoids downloading and installing an upgrade on top of the same version or a newer version of the same

software, which could cause serious problems for the user. Even if one considers this a "prerequisite" (and this is a stretch), it is a prerequisite to the installation of the upgrade, not a prerequisite applicable to individual files or sequences of commands in the upgrade. Applicants' use of prerequisite fields is associated with individual script objects. Therefore, some scripts objects may require the prerequisite and some may not. In particular, as a result of different prerequisites, a script may be executed out of sequence. None of these aspects is disclosed in *Chiles*.

Nor are the independent claims obvious over *Chiles*. As explained, *Chiles*' script amounts to a list of files to download (and run). All the work of the upgrade is done in the downloaded files. This is not much different from the prior art technique of creating customized installation software for each upgrade, except that it is easier to download multiple modules. Applicants' technique, by supported multiple upgrade (or installation) objects, each having its own prerequisites, allows greater flexibility . *Chiles* use of a single list of files to be downloaded does not teach or suggest the use of multiple upgrade objects, each having script commands of a pre-defined script instruction set which are compiled and executed by a script processor, and each having its own prerequisites.

In view of the foregoing, applicants submit that the claims are now in condition for allowance and respectfully request reconsideration and allowance of all claims. In addition, the

Examiner is encouraged to contact applicants' attorney by telephone if there are outstanding issues left to be resolved to place this case in condition for allowance.

Respectfully submitted,

SAMUEL D. DULL III, et al.

By:_____
Roy W. Truelson
Registration No. 34,265

Telephone: (507) 289-6256 (Office/Fax)
(507) 202-8725 (Cell)